📚 Plágium Ellenőrző Rendszer - Teljes Dokumentáció

# PLÁGIUM ELLENŐRZŐ RENDSZER - TELJES DOKUMENTÁCIÓ

## RENDSZER ÁTTEKINTÉS

A Plágium Ellenőrző Rendszer egy Robot Framework alapú automatizált megoldás, amely DOCX dokumentumokban keres ismétlődő tartalmakat és redundanciákat. A rendszer teljes mértékben automatikus, professzionális jelentéseket készít és emailben küldi el az eredményeket.

### FŐBB KÉPESSÉGEK

* 🔍 Automatikus plágium detektálás DOCX fájlokban
* 📊 Excel jelentések generálása
* 📧 Automatikus email küldés eredményekkel
* 🗄️ SQLite adatbázis a teljes előzmények tárolására
* 🎯 Három kategóriás értékelés: Rendben / Gyanús / Másolt
* 🔄 Batch feldolgozás több dokumentum egyidejű kezelésére

---

## RENDSZER ARCHITEKTÚRA

### FÁJL STRUKTÚRA

PlagiumEllenorzes/  
├── 🤖 PLG-00-main.robot # Fő Robot Framework teszt  
├── 📄 PLG-02-read\_docx.robot # DOCX beolvasó modul  
├── 📊 PLG-03-write-excel.robot # Excel export modul  
├── ⚙️ Plagium.config # Konfigurációs fájl  
├── 📚 libraries/ # Python modulok  
│ ├── 🐍 DocxReader.py # DOCX olvasó library  
│ ├── 📧 send\_email.py # Email küldő rendszer  
│ ├── 📊 excel\_export\_simple.py # Excel export engine  
│ ├── ⚙️ get\_config.py # Konfiguráció betöltő  
│ └── 🔧 plagium\_config.py # Python konfiguráció osztály  
├── 📂 resources/ # Robot Framework erőforrások  
│ ├── 🔑 keywords.robot # Kulcsszó definíciók  
│ └── 🔢 variables.robot # Változó definíciók  
├── 🗃️ test\_database.db # SQLite adatbázis  
├── 🐍 rf\_env/ # Python virtuális környezet  
└── 📊 results/ # Eredmény fájlok

### TECHNOLÓGIAI STACK

* 🤖 Robot Framework: Automatizálási keretrendszer
* 🐍 Python 3.8+: Háttér programozási nyelv
* 📄 python-docx: DOCX fájl feldolgozás
* 🗄️ SQLite: Beépített adatbázis
* 📊 openpyxl: Excel fájl generálás
* 📧 pywin32: Windows Outlook integráció
* 🗃️ DatabaseLibrary: Robot Framework adatbázis támogatás

---

## KONFIGURÁCIÓ

### Plagium.config FÁJL

# Email beállítások  
email=lovasz.otto@clarity.hu  
  
# Bementi könyvtár - ide kerüljenek a feldolgozandó DOCX fájlok  
input\_folder=d:\tmp  
  
# Kimeneti könyvtár - ide készülnek az Excel jelentések  
output\_folder=d:\tmp  
  
# Adatbázis beállítások  
database\_file=test\_database.db  
  
# Email tárgy beállítása  
email\_subject=Plagium Ellenorzes - Eredmenyek  
  
# Excel fájl prefix  
excel\_prefix=plagium\_eredmenyek  
  
# Debug mód (true/false)  
debug\_mode=false

### TESTRESZABHATÓ PARAMÉTEREK

|  |  |  |
| --- | --- | --- |
| **Paraméter** | **Leírás** | **Példa érték** |
| email | Címzett email cím | felhasznalo@company.hu |
| input\_folder | DOCX fájlok forrás könyvtára | C:\Documents\ToCheck |
| output\_folder | Excel jelentések célkönyvtára | C:\Reports\Output |
| database\_file | SQLite adatbázis fájl neve | plagium\_database.db |
| email\_subject | Email tárgy sablon | Plágium Jelentés - {dátum} |
| excel\_prefix | Excel fájlok elnevezési prefix | jelentes\_plagium |

---

## ADATBÁZIS STRUKTÚRA

### REDUNDANCIA TÁBLA

CREATE TABLE redundancia (  
 id INTEGER PRIMARY KEY AUTOINCREMENT,  
 status TEXT DEFAULT 'Rendben', -- Rendben/Gyanús/Másolt  
 file\_name TEXT NOT NULL, -- Fájl neve  
 file\_size INTEGER NOT NULL, -- Fájl méret (byte)  
 record\_date TEXT NOT NULL, -- Feldolgozás dátuma  
 record\_time TEXT NOT NULL, -- Feldolgozás időpontja  
 max\_ismetlesek\_szama INTEGER DEFAULT 0, -- Legnagyobb ismétlések száma  
 max\_ismetelt\_karakterszam INTEGER DEFAULT 0, -- Leghosszabb ismételt szöveg  
 overview TEXT DEFAULT '' -- Összefoglaló szöveg  
);

### HASHCODES TÁBLA

CREATE TABLE hashCodes (  
 hash\_value TEXT(100) PRIMARY KEY, -- SHA-256 hash érték  
 file\_name TEXT NOT NULL, -- Fájl neve  
 file\_path TEXT NOT NULL, -- Teljes fájl útvonal  
 created\_date TEXT NOT NULL, -- Létrehozás dátuma  
 created\_time TEXT NOT NULL, -- Létrehozás időpontja  
 line\_content TEXT, -- Sor tartalma  
 redundancia\_id INTEGER, -- Kapcsolat a redundancia táblával  
 FOREIGN KEY (redundancia\_id) REFERENCES redundancia(id)  
);

### REPEAT TÁBLA

CREATE TABLE repeat (  
 id INTEGER PRIMARY KEY AUTOINCREMENT,  
 file\_name TEXT NOT NULL, -- Aktuális fájl neve  
 source\_file\_name TEXT NOT NULL, -- Forrás fájl neve  
 redundancia\_id INTEGER, -- Kapcsolat a redundancia táblával  
 block\_id INTEGER NOT NULL, -- Blokk azonosító  
 line\_length INTEGER NOT NULL, -- Sor hossza  
 sum\_line\_length INTEGER DEFAULT 0, -- Összesített hossz  
 repeated\_line TEXT NOT NULL, -- Ismétlődő szöveg  
 created\_date TEXT NOT NULL, -- Létrehozás dátuma  
 created\_time TEXT NOT NULL, -- Létrehozás időpontja  
 FOREIGN KEY (redundancia\_id) REFERENCES redundancia(id)  
);

---

## PLÁGIUM DETEKTÁLÁSI ALGORITMUS

### HASH-ALAPÚ ÖSSZEHASONLÍTÁS

1. 📄 Dokumentum beolvasás: DOCX fájl szöveges tartalmának kinyerése
2. ✂️ Szöveg szegmentálás: Sorok és bekezdések szétbontása
3. 🔐 Hash generálás: SHA-256 hash értékek számítása minden sorhoz
4. 🔍 Összehasonlítás: Hash értékek összevetése a meglévő adatbázissal
5. 📊 Redundancia számítás: Ismétlődő tartalmak hosszának meghatározása

### KATEGORIZÁLÁSI SZABÁLYOK

if max\_ismetelt\_karakterszam < 300:  
 status = "🟢 Rendben" # Minimális redundancia  
elif 300 <= max\_ismetelt\_karakterszam < 1200:  
 status = "🟡 Gyanús" # Jelentős redundancia  
else:  
 status = "🔴 Másolt" # Kritikus redundancia

### ÉRTÉKELÉSI METRIKÁK

* 📏 Karakterszám alapú: Ismétlődő szövegrészek teljes hossza
* 🔢 Gyakoriság alapú: Ismétlődések számának figyelembevétele
* 📊 Százalékos arány: Redundancia az egész dokumentumhoz viszonyítva

---

## EMAIL RENDSZER

### AUTOMATIKUS KÜLDÉSI MECHANIZMUS

A rendszer háromszintű email küldési stratégiát alkalmaz:

outlook = win32com.client.Dispatch("Outlook.Application")  
mail = outlook.CreateItem(0) # Email objektum  
mail.To = recipient\_email  
mail.Subject = subject  
mail.HTMLBody = html\_content  
mail.Attachments.Add(excel\_file\_path)  
mail.Send() # 🚀 AUTOMATIKUS KÜLDÉS

# Ha a Send() nem működik  
mail.Save() # Piszkozatként mentés

# Részletes hibanapló generálás  
error\_report = generate\_error\_details()  
log\_to\_file(error\_report)

### EMAIL SABLON STRUKTÚRA

<!DOCTYPE html>  
<html>  
<head>  
 <style>  
 /\* Modern CSS stílusok \*/  
 .status-ok { color: #28a745; }  
 .status-suspicious { color: #ffc107; }  
 .status-copied { color: #dc3545; }  
 table { border-collapse: collapse; width: 100%; }  
 th, td { border: 1px solid #ddd; padding: 8px; }  
 </style>  
</head>  
<body>  
 <h2>🔍 Plágium Ellenőrzés Eredményei</h2>  
  
 <!-- Összesítő statisztikák -->  
 <div class="summary">  
 <h3>📊 Összesítés</h3>  
 <p>Összesen: {total\_docs} dokumentum</p>  
 <p class="status-ok">🟢 Rendben: {clean\_docs}</p>  
 <p class="status-suspicious">🟡 Gyanús: {suspicious\_docs}</p>  
 <p class="status-copied">🔴 Másolt: {copied\_docs}</p>  
 </div>  
  
 <!-- Részletes táblázat -->  
 <table>  
 <thead>  
 <tr>  
 <th>Státusz</th>  
 <th>Fájl név</th>  
 <th>Max redundancia</th>  
 <th>Dátum</th>  
 </tr>  
 </thead>  
 <tbody>  
 <!-- Dinamikus sorok -->  
 </tbody>  
 </table>  
</body>  
</html>

---

## FUTTATÁSI MÓDOK

### 1️⃣ MANUÁLIS FUTTATÁS

# Robot Framework direktben  
rf\_env\Scripts\robot.exe PLG-00-main.robot  
  
# Vagy batch fájlon keresztül  
start.bat

### 2️⃣ BATCH FELDOLGOZÁS

\*\*\* Keywords \*\*\*  
Batch DOCX ellenőrzés  
 @{docx\_files}= List Files In Directory ${DOCUMENT\_PATH} \*.docx  
 FOR ${docx\_file} IN @{docx\_files}  
 ${redundancia\_id}= Fájladatok Rögzítése Redundancia Táblába ${docx\_file}  
 DOCX Beolvasás Teszt  
 END

### 3️⃣ ÜTEMEZETT FUTTATÁS

# Windows Task Scheduler integráció  
schtasks /create /tn "Plagium Check" /tr "C:\PlagiumEllenorzo\start.bat" /sc daily /st 09:00

---

## KIMENETI FORMÁTUMOK

### EXCEL JELENTÉS

A generált Excel fájl a következő oszlopokat tartalmazza:

|  |  |  |
| --- | --- | --- |
| **Oszlop** | **Leírás** | **Példa érték** |
| ID | Egyedi azonosító | 42 |
| Státusz | Kategorizálás | 🟢 Rendben |
| Fájl név | Dokumentum neve | diplomamunka.docx |
| Fájl méret | Méret bájtokban | 2,456,789 |
| Dátum | Feldolgozás dátuma | 2025-08-25 |
| Idő | Feldolgozás időpontja | 14:32:15 |
| Max ismétlés | Legnagyobb ismétlési szám | 7 |
| Max karakterszám | Leghosszabb redundancia | 1,847 |
| Áttekintés | Összefoglaló megjegyzés | Jelentős átfedések |

### ROBOT FRAMEWORK JELENTÉSEK

* 📄 log.html: Részletes végrehajtási napló
* 📊 report.html: Összefoglaló jelentés
* 📝 output.xml: Strukturált XML kimenet

---

## HIBAELHÁRÍTÁS

### GYAKORI HIBÁK ÉS MEGOLDÁSOK

HIBA: Email küldés sikertelen  
MEGOLDÁS:  
1. Ellenőrizd, hogy az Outlook fut-e  
2. Konfiguráld az Outlook biztonsági beállításait  
3. Engedélyezd a COM objektumok használatát  
4. Ellenőrizd az internet kapcsolatot

HIBA: Database lock error  
MEGOLDÁS:  
1. Zárd be az összes SQLite kapcsolatot  
2. Ellenőrizd a fájl jogosultságokat  
3. Újraindítás után próbálkozz újra

HIBA: Corrupt DOCX file  
MEGOLDÁS:  
1. Ellenőrizd a fájl sértetlenségét  
2. Nyisd meg Word-ben és mentsd újra  
3. Konvertáld más formátumból

### DEBUG MÓDOK

# Plagium.config módosítás  
debug\_mode=true

import logging  
logging.basicConfig(level=logging.DEBUG)

robot --loglevel DEBUG PLG-00-main.robot

---

## BIZTONSÁGI MEGFONTOLÁSOK

### ADATVÉDELEM

* ✅ Helyi adatkezelés: Minden adat a helyi gépen marad
* ✅ Titkosított hash-ek: SHA-256 kriptográfiai hash használata
* ✅ Hozzáférés-szabályozás: Fájlrendszer szintű jogosultságok
* ✅ Audit napló: Teljes műveleti előzmények

### HASH ALGORITMUS BIZTONSÁG

import hashlib  
  
def generate\_secure\_hash(content):  
 """  
 SHA-256 alapú biztonságos hash generálás  
 - Kriptográfiailag biztonságos  
 - Ütközés-rezisztens  
 - Nem visszafejthető  
 """  
 return hashlib.sha256(content.encode('utf-8')).hexdigest()

### EMAIL BIZTONSÁG

* ✅ Outlook integráció: Biztonságos COM objektum használat
* ✅ Hitelesítés: Windows beépített hitelesítés
* ✅ Titkosítás: Email forgalom TLS titkosítással

---

## TELJESÍTMÉNY OPTIMALIZÁLÁS

### FUTÁSI IDŐ OPTIMALIZÁLÁS

# Batch méret beállítás  
BATCH\_SIZE = 1000  
  
# Memória hatékony feldolgozás  
def process\_large\_document(file\_path):  
 with open(file\_path, 'r', encoding='utf-8') as f:  
 while True:  
 lines = f.readlines(BATCH\_SIZE)  
 if not lines:  
 break  
 process\_batch(lines)

### ADATBÁZIS TELJESÍTMÉNY

-- Indexek létrehozása  
CREATE INDEX idx\_redundancia\_status ON redundancia(status);  
CREATE INDEX idx\_hashcodes\_file ON hashCodes(file\_name);  
CREATE INDEX idx\_repeat\_redundancia ON repeat(redundancia\_id);  
  
-- Vacuum és analyze  
VACUUM;  
ANALYZE;

### MEMÓRIA MENEDZSMENT

* ✅ Streaming feldolgozás: Nagy fájlok darabos beolvasása
* ✅ Garbage collection: Python automatikus memória felszabadítás
* ✅ Connection pooling: Adatbázis kapcsolatok újrahasznosítása

---

## FRISSÍTÉSEK ÉS KARBANTARTÁS

### KOMPONENS FRISSÍTÉSEK

# Robot Framework frissítés  
pip install --upgrade robotframework  
  
# Python csomagok frissítése  
pip install --upgrade python-docx openpyxl pywin32  
  
# Virtuális környezet újraépítése  
deactivate  
rmdir /s rf\_env  
python -m venv rf\_env  
rf\_env\Scripts\activate  
pip install -r requirements.txt

### ADATBÁZIS KARBANTARTÁS

-- Régi rekordok archíválása (90 napnál régebbiek)  
CREATE TABLE redundancia\_archive AS  
SELECT \* FROM redundancia  
WHERE record\_date < date('now', '-90 days');  
  
DELETE FROM redundancia  
WHERE record\_date < date('now', '-90 days');  
  
-- Adatbázis optimalizálás  
VACUUM;  
REINDEX;

### KONFIGURÁCIÓ BACKUP

# Automatikus backup script  
$date = Get-Date -Format "yyyyMMdd\_HHmmss"  
Copy-Item "Plagium.config" "backup\Plagium.config.$date"  
Copy-Item "test\_database.db" "backup\test\_database.db.$date"

---

## FEJLESZTŐI ÚTMUTATÓ

### FEJLESZTŐI KÖRNYEZET BEÁLLÍTÁSA

# 1. Repository klónozása  
git clone https://github.com/your-repo/plagium-checker.git  
  
# 2. Virtuális környezet létrehozása  
python -m venv dev\_env  
dev\_env\Scripts\activate  
  
# 3. Fejlesztői csomagok telepítése  
pip install -r requirements-dev.txt  
  
# 4. Pre-commit hooks beállítása  
pre-commit install

### TESZTELÉSI STRATÉGIA

\*\*\* Test Cases \*\*\*  
Unit Test - Hash Generation  
 ${hash}= Generate Hash "test content"  
 Should Match Regexp ${hash} ^[a-f0-9]{64}$  
  
Integration Test - Email Sending  
 ${result}= Send Test Email test@example.com  
 Should Be Equal ${result} success  
  
Performance Test - Large Document  
 ${time\_start}= Get Time epoch  
 Process Document large\_test\_file.docx  
 ${time\_end}= Get Time epoch  
 ${duration}= Evaluate ${time\_end} - ${time\_start}  
 Should Be True ${duration} < 300 # Max 5 perc

### KÓDMINŐSÉG BIZTOSÍTÁS

# .github/workflows/quality.yml  
name: Code Quality  
on: [push, pull\_request]  
jobs:  
 test:  
 runs-on: windows-latest  
 steps:  
 - uses: actions/checkout@v2  
 - name: Setup Python  
 uses: actions/setup-python@v2  
 with:  
 python-version: '3.9'  
 - name: Install dependencies  
 run: pip install -r requirements.txt  
 - name: Run Robot Framework tests  
 run: robot --outputdir results tests/  
 - name: Code coverage  
 run: coverage run -m robot tests/

---

## TÁMOGATÁS ÉS KÖZÖSSÉG

### TÁMOGATÁSI CSATORNÁK

* 📧 Email támogatás: support@plagium-checker.com
* 💬 GitHub Issues: GitHub Issues oldal
* 📖 Dokumentáció: Online wiki
* 🎥 Videó útmutatók: YouTube channel

### KÖZREMŰKÖDÉS

# Közreműködési irányelvek  
  
## Pull Request folyamat  
1. Fork-old a repository-t  
2. Hozz létre feature branch-et  
3. Commitold a változásokat  
4. Írj teszteket az új funkcióhoz  
5. Küldj Pull Request-et  
  
## Kód stílus  
- Python: PEP 8 standard  
- Robot Framework: hivatalos style guide  
- Commit üzenetek: Conventional Commits format

---

## CHANGELOG ÉS VERZIÓKEZELÉS

### AKTUÁLIS VERZIÓ: v2.1.0

* ✅ ÚJ: Háromszintű email küldési rendszer
* ✅ JAVÍTÁS: Windows path escape karakterek kezelése
* ✅ OPTIMALIZÁLÁS: Libraries könyvtár szervezés
* ✅ FEJLESZTÉS: Tisztított logging rendszer
* ✅ ÚJ: Outlook COM automatikus email küldés
* ✅ ÚJ: Excel export funkció
* ✅ JAVÍTÁS: SQLite adatbázis optimalizálás
* ✅ FEJLESZTÉS: Batch feldolgozás támogatás
* ✅ ÚJ: Kategorizálási rendszer (Rendben/Gyanús/Másolt)
* ✅ ÚJ: Hash-alapú redundancia detektálás
* ✅ JAVÍTÁS: DOCX fájl beolvasás stabilitás

---

## JÖVŐBELI FEJLESZTÉSEK

### ROADMAP

* 🔮 AI-alapú szöveg analízis: GPT integráció fejlettebb detektáláshoz
* 📱 Web interfész: Browser-alapú felhasználói felület
* 🔗 API végpontok: REST API a külső integrációkhoz
* ☁️ Cloud támogatás: Azure/AWS adatbázis integráció
* 📊 Fejlett riportok: Power BI/Tableau kapcsolat
* 🔄 Real-time monitoring: Élő dashboard a feldolgozásokhoz
* 🤖 Machine Learning: Minta-alapú plágium felismerés
* 🌐 Multi-platform: Linux és macOS támogatás
* 📧 Fejlett értesítések: Slack/Teams integráció

---

## LICENC ÉS JOGI INFORMÁCIÓK

### SZOFTVER LICENC

MIT License  
  
Copyright (c) 2025 Plágium Ellenőrző Rendszer  
  
Permission is hereby granted, free of charge, to any person obtaining a copy  
of this software and associated documentation files (the "Software"), to deal  
in the Software without restriction, including without limitation the rights  
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
copies of the Software, and to permit persons to whom the Software is  
furnished to do so, subject to the following conditions:  
  
The above copyright notice and this permission notice shall be included in all  
copies or substantial portions of the Software.

### ADATVÉDELMI NYILATKOZAT

* ✅ Adatkezelés: Minden adat helyben marad
* ✅ Harmadik fél: Nincs adatmegosztás
* ✅ Tárolás: Lokális SQLite adatbázis
* ✅ Törlés: Felhasználó teljes kontrollja

---

## KAPCSOLAT

### FEJLESZTŐI CSAPAT

* 🧑‍💻 Főfejlesztő: [Név] - lead@plagium-checker.com
* 🔧 DevOps: [Név] - devops@plagium-checker.com
* 📋 Projektmenedzser: [Név] - pm@plagium-checker.com

### VÁLLALATI INFORMÁCIÓK

Plágium Ellenőrző Rendszer Kft.  
1234 Budapest, Példa utca 42.  
Adószám: 12345678-2-41  
Email: info@plagium-checker.com  
Telefon: +36-1-234-5678

---

📝 Ez a dokumentáció a Plágium Ellenőrző Rendszer v2.1.0 verziójához készült.

🔄 Utolsó frissítés: 2025. augusztus 25.

✨ Készítette: Robot Framework automatizált dokumentáció generátor